Chapter 9

Software Licenses

Software licensing is a confusing and broad topic, but the issues surrounding it affect us every day. Virtually every piece of software sold today requires a user to click "I Agree" to some text before allowing installation. The text is usually lengthy, legal sounding, and difficult to understand. The vast majority of users blindly click "I Agree" without bothering to read any portion of the text. To what exactly is a user agreeing? Can anything at all be put into the text? Is it possible a user could be agreeing to something ridiculous? What laws or legal standards control this process? What are companies entitled to protect in their software, and what rights do users have to use the software in the manner they choose? What recourse does a consumer have should something go wrong in the software?

Broadly speaking, two kinds of property have been developed throughout history: physical property and intellectual property. With physical property, such as a chair or a toaster, society is most concerned with safety. Manufacturers are expected to take reasonable precautions to insure the reliability and safety of their products. With intellectual property, such as a story or a song, society is most concerned with copying and distribution. In order to encourage the continued creation of new intellectual property, society expects the creators to be reasonably compensated and hence protected from unauthorized reproductions.

The problem with software is that it has aspects of both physical and intellectual property, and additional aspects that are completely new. For software that controls parts of an automobile, or a microwave, or a heart pacemaker, there are obvious safety issues. For software designed for entertainment, there are obvious copying and distribution issues. Completely new

issues have been raised by the practices of patching, updating, and maintenance of software. While these practices are currently recognized as the most practical and efficient methods for installing and maintaining software, they have no precedent. For example, consider allowing an automobile manufacturer the right to leave a key underneath the chasis of every car produced, for "authorized access" for maintenance. Should the manufacturer be allowed to access the car at any time, for safety updates, for advertising, or for tracking? While this sounds absurd, this is analagous to the backdoors and automated update procedures commonly used today by desktop software. Because there is no analagous precedent, society is trying to figure out how to cope with these new issues.

Software could be said to be in its "wild west" period of law enforcement. The existing laws that were originally designed for physical and intellectual property have shortcomings, gaps, and perhaps even contradictions when they are applied to software. As society tries to find methods to regulate software, new laws and policies are being developed at a rapid rate. Examples include the Digital Millenium Copyright Act (DMCA) and the Uniform Computer Information and Technology Act (UCITA). This chapter discusses several of these laws and the surrounding issues.

Meanwhile, while commercial software licensing is lost in the quagmire of "shrinkwrap licensing", another movement is taking place in "open source" software. What exactly is open source software? Can it be sold, or is it always free? How is it protected? What are the legal implications for making software available open source?

This chapter explores all these issues. The goal is to understand the concepts surrounding software licensing. A computing professional is very likely to encounter licensing issues during the course of his or her career. An understanding of these issues can help a developer decide what course to take in writing, adapting, and/or protecting software that is to be sold or otherwise made public.

9.1 Intellectual property protection

Intellectual property is about ideas, and the protection of those ideas, in order to encourage clever people to come up with new ideas. Inventions and innovation, the creative expression of music and art, the production of a story, these are all well-regarded things that benefit society and enhance

our lives. In order to promote the pursuit of these things, society has long adopted policies to reward those with new ideas. Monetary gain has always been the prime motivator. The prestige that goes with being known for a new idea is rewarding, but to live one must have money. Money supports the innovator while he or she pursues the new ideas that enrich society.

In older times, it was common for an artist to be supported by a wealthy patron. A composer or playwright could also make money through performances, where fees were collected from those in attendance. It was difficult to copy and disseminate works of knowledge or art. Since innovators were otherwise compensated, society was not concerned with providing extra compensation to innovators by way of protecting their ideas. For example, if someone other than the oritinal author put on a play, society did not much care if the original author was compensated. Most societies were interested in seeing new ideas by any means possible, because distribution was so difficult.

As the means for copying and distribution became more advanced, things changed. Society began to recognize the need to protect ideas, and those who made a business of new ideas. Although the term was not used until the mid-18th century, and not popular until the late 20th century, the era of intellectual property protection was begun. Intellectual property (IP) is a broad term, covering the legal distinctions and definitions surrounding the protection of ideas and those whose business is new ideas. This section discusses the classic IP protection schemes, and how they have evolved to eventually address software.

9.1.1 Patent

One of the two earliest forms of IP protection in history is the patent. A patent is given by a government to an inventor, giving the inventor the exclusive right to make and sell products based on the invention. In return, the inventor must disclose to the government (and hence, the public) exactly how to make the invention. This relationship is useful in both directions. It encourages the public disclosure of new ideas, so that the public may benefit. It also provides for the inventor, so that competitors can not profit by copying the invention. While the original inventor may make products based upon the patent, it is also common for the inventor to license the patent for a fee to others for production.

Patents were first seen in Europe in the 15th century, and became a widespread concept by the 17th century. The United States has had var-

ious patent laws since its inception, starting with the Patent Act of 1790. Early patents typically covered ideas that are mechanical in nature, such as methods for milling and machining. Chemical processes have also long been recognized as patentable ideas. However, as technology has progressed, the scope of patentable ideas has evolved. The biggest challenge in patent law today is software.

Prior to the 1980's, the United State Patent and Trademark Office[1] (USPTO) disallowed all patents involving software. The office had long held that mathematical and scientific truths could not be patented. It regarded software and algorithms as merely a form of mathematics. The line became blurred when software was used to control a machine to complete a process. If the machine could not complete the process without the software, and was otherwise innovative, then should not the total invention, even if a substantial part is software, be patentable? In a landmark case in 1981 (Diamond v. Diehr), the U.S. Supreme Court ruled that an invention fitting this category is patentable. Over time, computers have largely replaced mechanical controls in complex machines, because of the increased precision and speed. Many modern products could not be operated without computer-based control. With this trend, the idea of patenting a machine that uses software for control has become commonly accepted.

However, this also opened the door for patents involving software only. Patents have always been available for parts of processes. It is common for a patent to build upon an existing patent, improving an existing process in some manner. If software can be part of a patent, and if a new piece of software can be written that improves the overall process, then should not the new piece of software be patentable? Another confusing distinction is the requirement that a patent involving software include a machine that the software controls. What if the machine that is being controlled is a computer? For example, imagine a piece of software that makes a computer

[1]Throughout this chapter, the discussion will be largely framed around U.S. laws regarding intellectual property. The use of U.S. law to demonstrate concepts is not intended to promote U.S. law as a standard, or to slight the laws of any other country. They simply happen to be those with which this author is most familiar. In most cases similar problems are being addressed throughout the world. The World Trade Organization (WTO) and its members are currently pushing for more uniform intellectual property laws throughout the world, but those efforts tend to wait until a wide consensus is achieved. Many of the issues discussed in this chapter are currently controversial, and so are not yet widely standardized.

more efficient in any computation; should that software be patentable? One can also argue that all software must run on a machine, namely a computer. Can an implementation of the software on any computer suffice to satisfy the requirement for a physical implementation of the patent?

Throughout the 1980's and 1990's, a number of cases continuously challenged the boundaries of patent law regarding software. By the mid 1990's patents were being granted for software irrespective of any hardware that it required, and whether or not it controlled a machine unrelated to computation. At the time of this writing, there is still a great debate as to the proper scope of patent law regarding software. It is unclear whether the protection of software via patents benefits society in the manner originally intended by patent law. Software has become so widespread that a patent for a crucial piece of software can conceivably affect a vast number of products, and thus become very profitable through licensing. Some people argue that software patents protect these important innovations, thus spurring economic growth in the software industry. Others argue that software patents stifle innovation by unnecessarily restricting clever new uses of patented software. The long-term future of software patentability remains unclear.

9.1.2 Copyright

In its original form, copyright referred literally to the right to copy, meaning the right to produce and sell copies of an artistic work. First applied to books and writing, copyright has been expanded to other forms of creative endeavor, including music, pictures and film. It has evolved to include not only the right to copy verbatim the original work, but also to control the derivation of works based upon the original, and to protect against imitation that would devalue the original work. It differs from a patent in that a copyright protects the expression of an idea, rather than the idea itself. For example, a patent could be obtained for a locomotion process based upon riding turtles, and only the patent holder could produce and sell products based upon that idea. Thus, the idea of a turtle-riding locomotion system is patentable. In contrast, a copyright could be obtained for a story about characters that ride turtles, and only the copyright holder could perform and sell that story. However, other stories could be written about characters riding turtles, so long as they used different characters and did not so closely resemble the original story so as to be considered an imitation. Thus, an artistic expression of turtle riding is subjet to copyright.

The concept of copyright is a result of the invention of the printing press in the 15th century. Prior to its invention, the copying of a written work was an expensive process, in terms of the required time and skill. With the printing press, the copying of books became more practical. In order to derive profits from writing a book, an author could contract a printer to produce copies, sharing the profits of each copy of the work that was sold. Both parties gained, and society was enriched through the wider availability of books. Governments therefore supported these contracts, and became involved in their regulation. Typically, a copyright is registered with a government, which enforces its protection. Some governments recognize a copyright even without explicit registration; the copyright is assumed to exist upon the creation of an intellectual work.

In the United States, copyright law was established with the 1790 Copyright Act. It was revised several times, most notably by the 1976 Copyright Act. This update provided copyright protection for several communication mediums that had appeared during the 20th century, in particular radio, sound recordings, television and film. Not long after, copyright law was expanded to include protection for software. The protection afforded by copyright has been used to protect the look and feel of a program, meaning its user interface. Copyright can be used to prevent a competitor from imitating the user interface of an established product. However, this protection has turned out to be limited in practice because it is usually possible to write software that accomplishes the same tasks with a different user interface. Copyright is most often used for software as it was applied originally to books: to control its copying and sale.

A copyright serves two purposes. First, it defines the rights of the copyright holder, as follows:

- the right to produce and sell copies of a work;
- the right to create derivative works (sequels, adaptions in form different from the original);
- the right to perform or display the work publicly;
- the right to sell or assign these rights to another.

Although the primary motivation is to control revenue generated by the work, there are other motivations. A copyright helps the holder maintain

the reputation or public image of a work. It can also serve as a measure of production or industry.

The second purpose of a copyright is to explicitly allow specific practices that might otherwise be construed as prohibited by the copyright. These practices include:

- the right to re-sell a legally obtained copy to a third party;
- the practice of reproducing part of a work for the purpose of public criticism or review;
- the practice of copying a work for archival purposes, for example as a backup or to a more permanent medium;
- the practice of copying a work for educational or research purposes.

In U.S. copyright law, these practices are not explicitly defined (other than the right of "first sale", which specifically allows re-sale). Rather, they are protected under a set of conditions called "fair use" that a court is supposed to use to evaluate the potential for loss to the copyright holder. As these practices typically benefit society and result in minimal monetary loss to the copyright holder, they are commonly upheld as legal. One must remember that the intent of copyright (and patent) law is to *balance* the royalties given to innovators against the benefit to society.

9.2 Technology-related laws for IP protection

Technology has played a large role in simplifying the copying and distribution of artistic and intellectual works. This is particularly true of the newer communication mediums, including radio, television, film, and the internet. While technology enables the copying necessary for the legal distribution and sale of a work, it also enables opportunities for violating copyrights. As a result, society has looked for new ways to enforce IP protection. This section explores some of the laws that have been passed with respect to IP protection in and through technology. Although our primary interest in this text is software, many of the relevant issues have precedents in other media, particularly music and film, which are relatively older than software.

Several themes can be seen in these laws. First, there is an expectation that technology can somehow be used to protect IP, and make unauthorized

copying prohibitively difficult. Protection schemes have been devised for several media, such as the serical copy management system (SCMS) used for digital audio tape, and the content scramble system (CSS) and content protection for recordable media (CPRM) system used for digital versatile discs (DVDs). Some laws have been passed requiring these types of protection schemes be built into every device that operates on a given medium. The expectation is that the technology will help enforce IP protection.

A second theme seen in recent laws is the inclusion of a provision that makes circumventing IP protection illegal. It is reasonable to suppose that any technology designed for protection can be thwarted by another technology. By making the "cracking" of a protection technology illegal, the laws provide for additional penalties to be imposed upon those who thwart a protection scheme. It is interesting to note that the act of unauthorized copying is already illegal, so that penalizing one who cracks a protection scheme while in the process of unauthorized copying is supplementary. A precedent for this additional penalty is the act of penalizing a thief for cracking open a safe while in the process of a burglary.

A third theme seen in recent laws concerns a form of a taxation on blank media. Even with new technologies to protect IP, and new penalties to protect those technologies, there is a recognition that some amount of unauthorized copying is going to happen. Therefore, the holders of copyrights seek to recoup losses by sharing in the profits of all blank media sold, by assuming that some portion of those media are being used for illicit copying. A small percentage of the profit from the sale of each blank medium goes to the copyright holders. While this seems like a good idea in principle, in practice it has difficulties. The vast number of copyright holders and copyrighted materials makes it difficult to equitably disseminate any revenue collected through taxation of blank media.

The following sections discuss some of the legislation and history that demonstrates these concepts.

9.2.1 Betamax case

In 1984, a legal battle was fought between Sony Corporation and Universal City Studios, Inc. The case concerned home video recorders, produced by Sony (among others), and television shows produced by Universal Studios (among others). Universal argued that the home video recorders facilitated unauthorized copying and should be prohibited. The U.S. Supreme Court

eventually ruled that the use of a recorder for *time shifting*, which is the watching of a television show at a time other than when originally broadcast, is a non-infringing and fair use. Since the recorder has a fair use, and is marketed with that intention, it was deemed legal. This set the precedent for allowing the sale of recording equipment to consumers for use in the home. It also set the precedent that a recording technology demonstrate it has a non-infringing use in order to be deemed legal.

After the case was settled in court, the film industry lobbied the U.S. Congress to pass lesiglation concerning using home recording equipment for illicit copying. By this time, the home video recorder had become commonplace, without a noticeable impact on piracy or copyright revenue. Therefore Congress rejected the arguments and as a compromise, passed a law requiring a roylaty be collected on the sale of all blank video recording tapes. This set the precedent of collecting a royalty on recording equipment and media to be paid to copyright holders.

9.2.2 Audio Home Recording Act (AHRA)

Digital audio tape was developed in the 1980's, providing a notable improvement over analog audio tape. Using an analog medium, noise is introduced during the copying process. Even with the highest quality equipment, a copy is always somewhat diminished compared to the original. In contrast, digital copying inserts no additional noise, providing an exact reproduction of the original. An analog medium also typically deteriorates over time, so that playback sounds different, usually worse, the older a recording gets. A digital medium always sounds the same so long as the medium is readable.

The Recording Industry Association of America (RIAA) lobbied fiercely against the sale of digital audio recording equipment to consumers. The association feared that it would lead to widespread piracy. In 1992 the Audio Home Recording Act (AHRA) amended the US Copyright Act by adding provisions for digital audio recording devices and media. Those provisions include:

1. Digital tape recorders must use the serial copy management system (SCMS), which limits copying beyond one generation.

2. The manufacture and sale of devices that circumvent SCMS is prohibited.

3. A royalty is collected for copyright holders from the sale of all digital tape recorders and blank digital tapes.

Digital audio tape is clearly a superior product when compared to analog audio tape. However, the RIAA delayed releasing copyrighted material on digital audio tape while lobbying for the AHRA. By the time the law was passed, the compact disc audio technology had been commercialized. Because a compact disc is superior to digital audio tape, digital audio tape never achieved any popularity. In effect, digital tape missed its brief opportunity for marketing because of this delay, and has since been relegated to a niche market. However, the AHRA set the precedent of requiring technology to enforce IP protection, and making circumvention of that technology illegal.

9.2.3 No Electronic Theft (NET) Act

In 1994, a new type of copyright infringement came to the attention of society. A university student named David LaMacchia had been operating an online computer service. Using that service, he had encouraged the posting of copies of copyrighted software, including games and office applications. Once posted, a piece of software could be copied by any user accessing the service. In effect, LaMacchia's service facilitated the illicit copying of copyrighted software.

In seeking to prosecute LaMacchia, the U.S. courts were faced with a problem. LaMacchia had not profited, or sought to profit, from the illicit copying. Without a motive of profit, he had not broken existing copyright law. The courts recommended that Congress consider this issue and pass legislation "closing the loop-hole" regarding the facilitation of copyright infringement using the internet. In 1997, the No Electronic Theft (NET) Act was passed in response. The NET Act made it a crime to engage in noncommercial copyright infringement. Although the original targeted media was software, the NET Act also applies to music and other media and has since been used in cases involving internet-based file sharing services.

9.2.4 Digital Millenium Copyright Act (DMCA)

The Digital Millenium Copyright Act (DMCA), passed in 1998, makes it illegal to circumvent any technology designed to protect intellectual property. Thus, it mimics and supercedes some provisions of the AHRA, and essentially

extends that concept to apply to any digital medium. It also makes it a crime to produce or distribute equipment or technology designed to circumvent or thwart an IP protection scheme.

Since its enactment, the DMCA has been a source of controversy. One issue concerns exactly what constitutes IP protection. One case brought to the U.S. courts, Chamberlain v. Syklink (2004), claimed that a coding system embedded in an electronic garage door opener constituted an IP protection scheme. Although this claim was ultimately rejected, similar cases have been brought to the courts. Exactly what constitutes an IP protection technology or scheme is still open to debate.

A second issue raised by the DMCA concerns fair use. There are certain practices, typically deemed beneficial to society with minimal loss to copyright holders, that conflict with the DMCA. These practices include making copies for education, research, criticism and satire. At the time of this writing, it is unclear how the DMCA affects these practices. For example, research into improving IP protection technologies, that necessarily requires the systematic testing and cracking of protection schemes, may actually be hindered by the DMCA.

9.3 Consumer protection and transactions

The previous sections discussed laws concerning the protection of intellectual property. For software, this mainly concerns the rights of software developers. In this section we discuss the laws concerning the expectations of consumers. In 1972, the U.S. passed the Consumer Product Safety Act and established a commission to monitor product safety. The commission covers a wide variety of products, including chemicals and merchandise. For specific products requiring a deeper expertise, or having a wider impact, the U.S. has established additional monitoring agencies. Examples include automobiles (monitored by the National Highway Traffic Safety Administration), guns (monitored by the Bureau of Alcohol, Tobacco, Firearms and Explosives), and food and drugs (monitored by the Food Drug Administration). However, at the time of this writing, none of these agencies has the charter to specifically monitor the safety of computing software. One may surmise that society does not yet see a need for its monitoring, but given the explosive proliferation of software, one may also surmise that it is only a matter of time. There are some that argue that the "wild west" time of software

licensing will only end when the software industry is suitably monitored and held accountable.

In addition to safety and reliability, a typical consumer is affected by the laws governing the sales and transactions of software. The following discusses the relevant laws that address commercial transactions.

9.3.1 Uniform commercial code (UCC)

The Uniform Commercial Code (UCC) governs sales, leases and transactions in the U.S. One of its goals was to simplify business transactions by making the rules governing transactions common across all 50 states. Originally composed in 1951, it was written with the intent that it be updated over time to maintain consistency with the currently best-regarded business practices. It is thus constantly under revision. The UCC was intended to reasonably "fill in the gaps" on all transactions in which terms were otherwise unspecified. For example, if there is no specification of warranty or return, then the UCC provides common terms.

Concerning the sales and transactions of software, there are two notable principles in the UCC. First, the UCC discourages the use of legal formalities in conducting transactions. This helps streamline business transactions by avoiding, whenever possible, the use of lawyers and legal paperwork. Second, the UCC assumes that there is a difference between merchants, who make a business producing or selling a particular product, and consumers. Merchants are expected to have a deeper knowledge of the product, and therefore are held to a higher burden in explaining any ramifications of a transaction. It is clear that the modern use of software licenses by developers goes against both these principles. Since the 1990's, software licenses have been growing in length and legal complexity, and common consumers are typically overwhelmed by the explanation of terms contained in the licenses.

9.3.2 Uniform Computer Information Transactions Act (UCITA)

The Uniform Computer Information Transactions Act (UCITA) was first proposed in 1999 as an extension to the UCC. Its intent was to provide a set of rules governing the sale of computer software, access to computing services, and other transactions in computing information. It caused great controversy

in both the computing and legal professions, after numerous organizations and interested parties pointed out shortcomings and other problems in the act, particularly where it allowed questionable practices. It was subsequently revised in 2002, and passed in 2 of 50 U.S. states (Virginia and Maryland) in 2004. Probably its most important contribution has been to draw attention to new issues that software raises in sales and transactions. Issues include:

- Should a software developer be liable for loss or damages caused by product (software) flaws? The scope of such losses could potentially go far beyond anything seen in other products.
- Should software be a sold or licensed product? Is a copy of a piece of software owned, or leased? Software can require continuous updating and patching to remain secure, interoperable, and relevant. Should a software developer be required to provide this service, or does that arrangement more closely resemble a traditional lease?
- Does a consumer have the right to modify or reverse engineer how a piece of software works, when strictly for personal use? Consumers commonly modify and reverse engineer products, such as automobiles and clothing. However, when a consumer uses a third-party fuel injector to improve engine performance, the modification cannot be made instantly and freely available to the entire world.

These issues do not yet have satisfactory answers. UCITA contines to be debated within the legal and computing professions.

9.4 Licensing

Understanding all the background in intellectual property and relevant legislation, we can now entertain a suitable discussion of software licensing. Broadly speaking, a license is a contract. It expresses the terms of a transaction. The sale of software does not require a license; without a license the sale would be governed like all other common sales not making use of contracts. However, software developers have used licenses to include uncommon or non-standard terms, by writing them into software licenses that must be accepted as a condition to using the software. At the time of this writing, there are primarily two types of licenses applied to software. Broadly

speaking, commercial software licenses typically include terms that provide additional rights to the developer, or restrict the rights of the user. Free and open source licenses typically include terms that provide additional rights to the user, and give away some of the normally reserved rights of the developer. The following sections discuss each in more detail.

9.4.1 End user license agreement (EULA)

Commercial software is usually sold under an end user license agreement (EULA). Physical copies of software are commonly sold in boxes or cases sealed with shrink wrapping; hence, a EULA is sometimes referred to as a shrink-wrap license. The most usual display of a EULA to a consumer is during installation of the software. The EULA will be displayed on the computer screen to the user along with a message box requiring the user to select "I Agree" or "I Accept". There are no specified legal boundaries on what terms a EULA may contain. Since the early 1990's, EULAs have in general been expanding both in length and scope. At the time of this writing, EULAs commonly contain many questionable terms. Some of the more interesting but questionable terms include:

- **Leased, not owned.** According to this term, the copy of the piece of software is leased, not owned. This potentially conflicts with the right to re-sell a legally obtained copy under copyright law.

- **Terms revealed after purchase.** The terms of a shrink-wrap license are revealed after the sale has been completed. The consumer is not aware of the terms until installation. In selecting "I Agree" during installation, the user is signing a contract (and reading it for the first time) after the purchase has already been completed.

- **Terms can be changed.** Some licenses state that the terms can be changed, even after the user has "signed the contract" by selecting "I Agree". The limits or boundaries to change are typically not defined, and so could be anything. Such a term usually states that the license can be modified at any time, at the discretion of the software developer, through an internet-based public notice.

- **Limited use of product.** A license may seek to limit the use of the software product. For example, a license may restrict how a piece

of software can be configured to interact with other products, thus preventing its use with a competitor's product.

- **Prohibit criticism or review.** Some licenses include a term that prohibits criticism or certain types of review of the software. This conflicts with practices allowed by copyright law.

- **Spyware.** A piece of software may include functions that track how a piece of software is used, or the performance of the software. This information is sent through the internet to the developer, for purposes of improving the software or for marketing to the consumer. A license may include a term informing a user of the existence of these functions, and restricting the user's rights to disable or otherwise control them.

- **Backdoors.** In order to periodically update a piece of software, or to retrieve information about how the software is being used, or to monitor its license, software may be constructed with a backdoor. A backdoor allows remote access to a computer running the software. This potentially creates a security hazard, in that a compromised backdoor could provide remote access to a malicious third party. A license may include a term informing a user of the existence of a backdoor, but limit the user's rights to disable or otherwise control it.

- **No warranty.** Most software licenses include a provision expressing that the developer provides no warranty for the product. The term typically states that the software developer can not be sued for any damages or loss caused by using the software, even if the software contains faults that result in damage or loss.

- **Sold as is.** Even if a piece of software does not work, a term that states the software is sold "as is" precludes a consumer from seeking compensation.

- **Legal jurisdiction restricted.** A common tactic to prevent litigation is to state a term restricting legal jurisdiction to the locale of the software developer. Thus, if a consumer seeks to sue a software developer, it must be done in that jurisdiction.

- **Reverse engineering prohibited.** Developers seek to protect their software through terms prohibiting reverse engineering. This prevents

competitors from imitating their software by learning how it operates. However, software commonly makes use of proprietary file formats for storing data. By prohibiting the reverse engineering of those formats, a software developer can prevent competitor products from being interoperable, thus locking in a market. Such a term also prevents a single consumer from making modifications desirable for personal use.

- **Remote shutdown or removal.** One of the most controversial terms is one that allows a developer to remotely shut down or remove a piece of software after determining that a license has expired, or is otherwise invalid. Using a backdoor and monitoring functions, a developer may affect a copy of software in a manner that prevents its operation. Normally, society requires legal intervention before a lender or leaser can recoup property after a consumer defaults on the terms of the contract. With instant remote access through the internet, it is unclear what legal standards regulate this process.

If challenged in court, it is not clear if each of these terms would be upheld as legal with regard to existing copyright, consumer protection, and transaction law. But without a regulatory body or law that explicitly governs software licensing, it should be expected that software developers would write licenses under terms most favorable to the developers. With the gaps and limitations of existing laws as they apply to software, one can understand how this state of affairs has come to exist.

9.4.2 Free and open source licenses

Some software developers have released their software under licenses that actually give away the rights normally retained under copyright law. These developers believe that this policy encourages the development and ongoing support of better software. The basic idea is that software that is free of copyright encumbrances can be modified and redistributed by anyone. Therefore, existing software that contains bugs can be fixed by anyone, and redistributed to the public. New features can be added to existing software by anyone, and redistributed for free. New software can be developed that makes use of existing software, thus shortening development time. These things can be done without paying a royalty, or having to obtain the permission of the original developer, because the copyright has literally been given

away. For books, film, and music, this approach seems undesirable because a consumer does not modify or extend these products. Software is different, because the application of modifications, updates and extensions is the rule rather than the exception.

Obviously, these practices affect the traditional business model surrounding intellectual property. Historically the primary business has been the copying and distribution of works. If these actions are free, then how can a software developer make money? To date, the primary answer has been in service and support. While anyone can freely obtain a copy of a piece of software licensed in this manner, it must still be installed and operated. All software requires some level of support to facilitate these actions; this is especially true for complex software like operating systems and server software. A developer can charge for this support. In addition, should a user desire customized modifications, a developer is free to charge for customized programming work. Finally, it is also possible to charge for copying and distribution, even though a user may find a free alternative source for the same materials. Why would anyone pay for something from one supplier that is free from a second supplier? The first supplier might bundle the software in a manner that makes its installation and use easier, or supply additional supporting materials.

Successful free and open source software typically involves an extended development community. Once released to the public, users and programmers other than the original software developers are expected to contribute to the improvement of the software. The improvements, whether bug fixes, feature extensions, or major revisions, are expected to be cycled back to the original developers. The developers can then incorporate the work of the extended community into future releases. The developers benefit from the coding work, testing and debugging, and novel ideas for features and extensions provided by the extended community. The users benefit by having an impact upon the continued development of software that is presumably of use to them. Everything in the cycle is done freely.

Free and open source licenses include terms to regulate these practices. The terms most commonly used include the following:

- **Software can be freely copied and distributed.** This term gives away the primary right that the developer can claim through copyright law. The same purpose could be achieved more simply by giving the copyright to the public domain. However, other terms in the license

normally go beyond the simple release of copyright.

- **Source code must be publicly available.** In order to modify software, for example to fix a bug or add a feature, it is easiest to work with the original source code. Therefore a term is normally included that requires the developer to make the source code freely and publicly available. In is noteworthy that this places a burden on the developer. Normally, the burden is considered slight in comparison to the benefits of fixes and improvements made for free by the development community.

- **Software can be used in any manner.** This term gives up another tenet of a copyright, in that anyone can operate the software publicly, or charge for services related to use of the software. It allows people other than the original developers to profit from use of the software, without having to pay a royalty or obtain permission. This term is generally included to encourage the use of free or open source software by the business community.

- **Software can be modified.** This term also gives up a tenet of copyright, in that anyone can develop a derivative work based upon the software. Besides allowing modifications for personal use, the term allows the modifications to be made public. This allows for improvements and extensions made by the development community to go back to the original developers. It also creates the possibility of forking, where multiple groups develop a piece of software in different manners, based upon varying popular trends or needs.

- **Derivatives must include same terms.** In order to promote free and open source software, this term requires that any new software must be licensed under the same terms as the software from which it was derived. This is intended to force the extended development community to operate under the same terms as the original developers. This term is sometimes called the "copyleft", as a play on the word copyright. It shows the irony of using copyright law, which the license is intended to supplant, to enforce the provisions of the license.

The GNU General Public License (GPL) is the most widely used free software license. However, there are dozens of licenses written by a variety of

organizations that promote free and open source software. They largely agree on the above terms, but may modify them or add to them in some manner. The reason that these licenses are referred to alternately as "free" or "open source" stems from a desire by some proponents to gain acceptance in the business community. There was a fear that the phrase "free software" could imply poor quality (as "free product" can imply poor quality) to those unfamiliar with the practice. The phrase "open source" was created to present a friendlier image to the business community. In practice, licenses using either description make use of the same terms.

While free and open source licenses provide greater rights to the public, they do not address all the issues raised by software as a product. For example, these licenses commonly include terms expressing no warranty, and that software is provided as-is. If open source software is provided as part of a bundled service, for a charge, and something goes wrong in the software, is the provider liable? Does a developer have any responsibility in the quality or intent of software provided to the public, or can anything at all be distributed? These questions remain open issues.

9.5 The Future

Based upon the trends seen in patent and copyright law, it is possible to forecast some proposals that may appear in the future (and in fact may have appeared already in various forms, whether proposed legislation, journalistic articles, or editorials). Can an IP protection technology be patented, or an IP protection scheme be copyrighted? Compression/decompression routines have followed this path, and had a profound impact on standards and competition. Would the same path in IP protection benefit society?

Should royalties be collected on all computing media, including floppy and hard drives, USB and flash storage devices, and any other newly invented media? The expectation may be that some portion of those media will always be used for illicit copying. But even assuming royalties should be collected, could they be equitably and fairly distributed to the proper copyright holders?

The question is even more profound regarding the internet. Should the internet be taxed, with a portion of the proceeds going to copyright holders? Presumably, some portion of all internet traffic involves illicit copying. Of course, equitable collection and distribution of any taxes collected becomes

even more problematic when considered on a world-wide scale. And would the costs of managing such a system outweigh the benefits to copyright holders?

There will always be a need for entertainment-related software. Presumably this market is best served by traditional copyright law, and will continue to operate in that manner. However, has the primary software needed to run standard computing systems reached a level of maturity such that it should be overseen by a regulatory body? Does public safety require it? Services that achieve market saturation, such as electricity, water and sewer, and telecommunications, generally require government oversight. Would the same approach be beneficial for standard computing system software?

No one can predict the future; it is possible that in the near future many of these issues will be resolved, and the rights of software developers will be harmoniously and equally balanced against the benefits of software to society. However, it will require not only prudent lawmakers and intellectual property advocates, but also informed computing professionals. Those making a profession in the computing industry must be aware of the issues that affect society through their profession. Software licensing, and the myriad of evolving laws and practices that surround it, is in its infancy. It is likely than the readers of this text will live and work in a time that will have the greatest impact upon its maturation.